APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/028,899	12/20/2001	Robert Alan Reid	01 P 09444 US	2160

21171   7590   01/11/2008
STAAS & HALSEY LLP
SUITE 700
1201 NEW YORK AVENUE, N.W.
WASHINGTON, DC 20005

EXAMINER
BULLOCK JR, LEWIS ALEXANDER

ART UNIT	PAPER NUMBER
2195	

MAIL DATE	DELIVERY MODE
01/11/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

PTOL-90A (Rev. 04/07)

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
  Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on _22 October 2007_.

2a)☒ This action is **FINAL.**   2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) _1-9 and 11-20_ is/are pending in the application.

   4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) _1-9 and 11-20_ is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/ are: a)☐ accepted or b)☐ objected to by the Examiner.

   Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

   Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   a)☐ All b)☐ Some * c)☐ None of:

   1.☐ Certified copies of the priority documents have been received.

   2.☐ Certified copies of the priority documents have been received in Application No. _____.

   3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

   * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1)☐ Notice of References Cited (PTO-892)

2)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3)☐ Information Disclosure Statement(s) (PTO/SB/08)
    Paper No(s)/Mail Date _____.

4)☐ Interview Summary (PTO-413)
    Paper No(s)/Mail Date. _____.

5)☐ Notice of Informal Patent Application

6)☐ Other: _____.

DETAILED ACTION

Claim Rejections - 35 USC § 102

1.   The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that

form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

2.   Claims 1-8, 11-17 and 19 are rejected under 35 U.S.C. 102(b) as being

anticipated by FLECK (U.S. Patent 6,128,641).

As to claim 1, FLECK teaches a method of chained switching execution of data

processing tasks (routines / interrupt handlers) comprising: storing a second return

address corresponding to a second data processing task and a third return address

corresponding to a third data processing task in a register (via storing the CSA's for the

routines / interrupt handler as they are switched out) (fig. 2); a first data processing task

(routine / interrupt handler) executing a call (return) to a task switching function (context

switch control unit / instruction control unit); the task switching function selecting and

executing a return operation to the task of the second return address or the third return

address (via selecting the head of the PCX for restoring whether its the first task or the

second task context / pointer) (via returning from the now current CSA to a previously

executing context) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30;

col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28). Fleck also

teaches the current context resides in the general purpose and program state registers

of the processor and to save the current context and create a new one, the following

steps can be performed by the processor hardware: a context save area is taken from the free list, the current context is stored into the save area, the save area is added to the head of the previous context list wherein the steps are performed in connection with a function call, or with taking a trap or interrupt thereby allowing the called function or interrupt or trap handler free to modify the general registers and other processor state without destroying the context of the calling function or interrupt task (col. 1, line 60 – col. 2, line 6). To switch back from a called function or trap or interrupt handler and switch back to the previous context, the following steps are performed by the processor handler: the save area at the head of the previous context list is removed from that list, the current context is loaded from the save area just removed from the previous context list, and the save area is added to the free context save area list. These steps are performed as part of the function return instruction, or the instruction to return from an interrupt or trap handler. FLECK also teaches on figure 4 that the PCX has three contexts which would be associated with three previously executing tasks / interrupt handlers. Therefore, the teachings of FLECK teach an interrupt handler/task executing such that when it finishes it invokes a return operation and the context switch control unit switches back to the head of the PCX pointer, thereby switching context / loading the previously executing second task / interrupt handler, such that when it finishes it invokes a return operation and the context switch control unit switches back to the new head of the PCX pointer, thereby switching context / loading the now previously executing third task / interrupt handler wherein all of the tasks are different from one

another (via one is a caller, one is the called, one is an interrupt handler and some of

the tasks execute in different regions).

As to claim 11, FLECK teaches an apparatus for chained switching execution of

a tasks (routines / interrupt handlers) on a data processor (data processing unit)

comprising: a memory (memory) having a first storage location for storing a return

address corresponding to a second task and a second storage location for storing a

return address corresponding to a third task (via storing CSA on previous context save

area list of all routines / interrupts switched); an input (call) for receiving information

indicative of instructions of a task switching function that has been called by the first

task (via a return from an executing routine / interrupt handler); and a memory

management apparatus (via the context switch control unit) coupled to the input and the

memory, and responsive to the instruction information indicating a return instruction for

moving the return address corresponding to the task from the first storage location or

the second storage location to a register of the data processor to be executed by a

return operation to the task (via performing a context switch thereby returning to a

previous CSA at the head of the list and executing the task from where the context was

saved) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines

49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28). Fleck also teaches the current

context resides in the general purpose and program state registers of the processor and

to save the current context and create a new one, the following steps can be performed

by the processor hardware: a context save area is taken from the free list, the current

context is stored into the save area, the save area is added to the head of the previous

context list wherein the steps are performed in connection with a function call, or with

taking a trap or interrupt thereby allowing the called function or interrupt or trap handler

free to modify the general registers and other processor state without destroying the

context of the calling function or interrupt task (col. 1, line 60 – col. 2, line 6). To switch

back from a called function or trap or interrupt handler and switch back to the previous

context, the following steps are performed by the processor handler: the save area at

the head of the previous context list is removed from that list, the current context is

loaded from the save area just removed from the previous context list, and the save

area is added to the free context save area list. These steps are performed as part of

the function return instruction, or the instruction to return from an interrupt or trap

handler. FLECK also teaches on figure 4 that the PCX has three contexts which would

be associated with three previously executing tasks / interrupt handlers. Therefore, the

teachings of FLECK teach an interrupt handler/task executing such that when it finishes

it invokes a return operation and the context switch control unit switches back to the

head of the PCX pointer, thereby switching context / loading the previously executing

second task / interrupt handler, such that when it finishes it invokes a return operation

and the context switch control unit switches back to the new head of the PCX pointer,

thereby switching context / loading the now previously executing third task / interrupt

handler wherein all of the tasks are different from one another (via one is a caller, one is

the called, one is an interrupt handler and some of the tasks execute in different

regions).

As to claim 17, FLECK teaches a data processing apparatus, comprising: a data processing portion for executing tasks (data processing unit); a task switcher (context switch control unit) coupled to the data processing portion for switching from execution of a first task (routine / interrupt handler) to execution of a second task or a third task (routine / interrupt handler) (via a return operation), the task switcher including a memory (memory) having a first storage location for storing a return address corresponding to the second task or second storage location for storing a return address corresponding to a third task (via storing CSA on previous context save area list, thereby routines / interrupt handlers swapped out contexts are stored and saved to be restored), and an input for receiving information indicative of instructions of a task switching function that has been called by the first task (via a return operation to return execution to the previously executing context of a task or interrupt handler); a register coupled to the task switcher (data registers / address registers); and the task switcher including a memory management apparatus coupled to the input and the memory; and responsive to the instruction information indicating a return instruction for moving the return address from the first or second storage location to the register of the processor and the task corresponding to the address is restored (via performing a context switch thereby returning to a previous CSA) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28). Fleck also teaches the current context resides in the general purpose and program state registers of the processor and to save the current context and create a new one, the

following steps can be performed by the processor hardware: a context save area is taken from the free list, the current context is stored into the save area, the save area is added to the head of the previous context list wherein the steps are performed in connection with a function call, or with taking a trap or interrupt thereby allowing the called function or interrupt or trap handler free to modify the general registers and other processor state without destroying the context of the calling function or interrupt task (col. 1, line 60 – col. 2, line 6). To switch back from a called function or trap or interrupt handler and switch back to the previous context, the following steps are performed by the processor handler: the save area at the head of the previous context list is removed from that list, the current context is loaded from the save area just removed from the previous context list, and the save area is added to the free context save area list. These steps are performed as part of the function return instruction, or the instruction to return from an interrupt or trap handler. FLECK also teaches on figure 4 that the PCX has three contexts which would be associated with three previously executing tasks / interrupt handlers. Therefore, the teachings of FLECK teach an interrupt handler/task executing such that when it finishes it invokes a return operation and the context switch control unit switches back to the head of the PCX pointer, thereby switching context / loading the previously executing second task / interrupt handler, such that when it finishes it invokes a return operation and the context switch control unit switches back to the new head of the PCX pointer, thereby switching context / loading the now previously executing third task / interrupt handler wherein all of the tasks are different from one

another (via one is a caller, one is the called, one is an interrupt handler and some of the tasks execute in different regions).

As to claim 2, FLECK teaches the selecting step includes the task switching function selecting a first pointer (address / pointer) that points to a first area of memory (CSA) where the return address is stored (col. 5, lines 49-67; col. 6, lines 9-12; col. 6, lines 47-54).

As to claim 3, FLECK teaches the pointer selecting step includes updating a second pointer to point to the first pointer (via restoring a CSA from the list / read-write-modify operation) (col. 6, lines 21-30).

As to claim 4, FLECK teaches the updating step includes updating the second pointer from a status wherein the second pointer points to a third pointer to a status wherein the second pointer points to the first pointer, and wherein the third pointer points to a second area of memory (via restoring a CSA from the list / read-write-modify operation) (col. 6, lines 21-30).

As to claim 5, FLECK teaches a return address corresponding to the first data processing task is stored in the second area of memory (via the second portion of the CSA which stores a pointer to another CSA / via following the list to the previous CSA) (col. 5, lines 49-61; col. 7, lines 10-28).

As to claim 6, FLECK teaches the task switching function storing the third pointer (via context switching functions thereby restoring CSAs from the list) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28).

As to claim 7, FLECK teaches the task switching function deselecting a return address (pointer / address) corresponding to the first data processing task (via restoring from the linked list) (col. 2, lines 7-21).

As to claim 8, FLECK teaches saving a return address (pointer / address) corresponding to the first data processing task (via switching context based on call) (col. 4, lines 10-30), and executing the saving step in parallel with the call executing step (col. 4, lines 49-55).

As to claim 12, FLECK teaches memory includes a second storage location for storing a first pointer (address / pointer) which points to a first area of the memory (CSA) that includes the first storage location, the memory management apparatus responsive to the instructions information for selecting the first pointer (via chaining addresses of CSAs to the list or using the list restore chained addressed CSAs on the list) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6, line 47 – col. 7, line 28).

As to claim 13, FLECK teaches the memory management apparatus includes a

memory manager for maintaining a second pointer (address / pointer), the memory

manager responsive to the instruction (context switch / restore) information for updating

the second pointer to point to the first pointer in the memory (via updating the listed

chained addressed CSAs after appending or removing CSA's to the list / read-write-

modify operation) (col. 1, lines 27-40; col. 1, line 57 – col. 2, line 32; col. 4, lines 10-30;

col. 5, lines 49-67; col. 6, lines 17-30; col. 6, line 47 – col. 7, line 28).

As to claim 14, FLECK teaches the memory manager is operable for updating

the second pointer from a status wherein the second pointer points to a third pointer

stored at a third location in the memory to a status wherein the second pointer points to

the first pointer, and wherein the third pointer points to a second area of the memory

(via context switching functions thereby storing CSAs to the list) (col. 1, lines 27-40; col.

1, line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col.

6, line 47 – col. 7, line 28).

As to claim 15, FLECK teaches the second area of the memory (part of CSA /

part of list element) includes a fourth storage location which stores therein a return

address (address / pointer) corresponding to the first data processing task (CSA for

function / interrupt handler) (via switching context based on call) (col. 4, lines 10-30).

As to claim 16, FLECK teaches the memory manager is responsive to the

instruction information for storing the third pointer (address / pointer of subsequent listed

CSAs) in the third location of the memory (via context switching based on a call to store

the address of CSA on list and link with the previous CSA) (col. 1, lines 27-40; col. 1,

line 57 – col. 2, line 32; col. 4, lines 10-30; col. 5, lines 49-67; col. 6, lines 17-20; col. 6,

line 47 – col. 7, line 28).

As to claim 19, FLECK teaches the register is a program counter register

(register) (col. 3, lines 21-32).

Claim Rejections - 35 USC § 103

3.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set
> forth in section 102 of this title, if the differences between the subject matter sought to be patented and
> the prior art are such that the subject matter as a whole would have been obvious at the time the
> invention was made to a person having ordinary skill in the art to which said subject matter pertains.
> Patentability shall not be negatived by the manner in which the invention was made.

4.   Claim 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over FLECK

(U.S. Patent 6,128,641).

As to claim 18, FLECK substantially discloses the invention above. However,

FLECK does not teach the switcher includes TriCore data processor architecture.

Official Notice is taken in that a Tri-Core data processor architecture is well known in the

art and therefore would be obvious to one skilled in the art that the system of FLECK is

implemented in a Tri-Core architecture in order to reduce execution overhead.

5.   Claims 9 and 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over

FLECK (U.S. Patent 6,128,641) in view of Applicant's Admitted Prior Art (APA).

As to claim 9, FLECK substantially discloses the invention above. However,

FLECK does not teach the data processing task being a host task, disk task, or servo

task. APA teaches that a data processing task is one of a host task, a disk task and a

servo task of an optical drive control system (pg. 1, line 21 – 2, line 1). It would be

obvious to one skilled in the art at the time of the invention to combine the teachings of

FLECK with the teachings of APA in order to switch firmware tasks more quickly than

conventional microprocessors and microcontrollers (col. 4, lines 38-41; col .1, lines 27-

31).

As to claim 20, FLECK substantially discloses the invention above. However,

FLECK does not teach the data processing task being a host task, disk task, or servo

task. APA teaches that a data processing task is one of a host task, a disk task and a

servo task of an optical drive control system (pg. 1, line 21 – 2, line 1). It would be

obvious to one skilled in the art at the time of the invention to combine the teachings of

FLECK with the teachings of APA in order to switch firmware tasks more quickly than

conventional microprocessors and microcontrollers (col. 4, lines 38-41; col. 1, lines 27-

31).

Response to Arguments

6.   Applicant's arguments filed October 22, 2007 have been fully considered but they are not persuasive. Applicant's amendment attempted to change the claims to what the Examiner detailed could be a novel feature. However, Applicant used the alternative language of "or" which allows for both the prior interpretation and the new interpretations. For instance, regarding the language of storing return addresses to a first and second processing task in a register, there is no limitation detailing how this information is stored. It could very much be stored in the same manner that Fleck does by chaining the contexts. The language detailing the task switching function selecting the second **or** third return address and executing a return operation to that selected address reads upon the same teachings of Fleck that the last tasks pointer context is selected during a return operation and loaded such that that tasks information is restored. Applicant's invention allows for the bypassing of the last saved information and this would read over the prior art teaching. Claim language detailed below would achieve this. The other claims do not overcome the prior art of record as detailed above also. The examiner welcomes Applicant's call for further clarification, but feels that the claim language provided below better claims the uniqueness of Applicants invention over the prior art of Fleck.

Possible Allowable Claim Language

Claim 1 (Currently Amended) A method of chained switching execution of data

processing tasks, comprising:

executing a third data processing task until a task switching opportunity occurs;

when a task switch opportunity occurs, the third data processing task calling a

task switching function to store the context of the second task in a register;

executing a second data processing task until a second task switching

opportunity occurs;

when a second task switching opportunity occurs, the second data processing

task calling the task switching function to store the context of the second task linked

ahead of the context of the third task in the register;

executing a first data processing task;

storing a second return address corresponding to a second data processing task

and a third return address corresponding to a third data processing task in a register;

a the first data processing task executing a return call to a the task switching

function;

the task switching function selecting the second return address or the third return

address from the register based on the return call while maintaining the storage of the

second return address;

the task switching function executing a return operation to the data processing

task corresponding to the selected return address.

Cancel claim 7, since the return address to the first data processing task was never
stored **or change it to a second data processing task or make it depend on claim 8
since this is the first instance of saving the first data processing task**.

Claim 11 (Currently Amended) An apparatus for chained switching execution of tasks
on a data processor, comprising:

a memory having a first storage location for storing a return address
corresponding to a second task and a second storage location for storing a return
address corresponding to a third task <u>wherein the first storage location is linked ahead
of the second storage location</u>;

an input for receiving information indicative of instructions of a task switching
function that has been called by a first task;

a memory management apparatus coupled to said input and said memory, and
responsive to said instruction information indicating a return instruction for moving said
return address from said ~~first storage location or said~~ second storage location to a
register of the data processor <u>while maintaining the first storage location in memory</u>;
and

wherein said data processor executes a return operation to the task
corresponding to the return address stored in the register of the data processor.

Claim 17 (Currently Amended) A data processing apparatus, comprising:

a data processing portion for executing data processing tasks;

a task switcher coupled to said data processing portion for switching from execution of a first task to execution of a second task or a third task, said task switcher including a memory having a first storage location for storing a return address corresponding to the second task, and a second storage location for storing a return address corresponding to the third task <u>wherein the first storage location is linked ahead of the second storage location</u>, and an input for receiving information indicative of instructions of a task switching function that has been called by the first task;

a register coupled to said task switcher;

said task switcher including a memory management apparatus coupled to said input and said memory, and responsive to said instruction information indicating a return instruction for moving said return address from said ~~first storage location or said~~ second storage location to a register of the data processing apparatus <u>while maintaining the first storage location in memory</u>; and

the task switcher switching from execution of the first task to execution of the task corresponding to the return address stored in the register of the data processing apparatus.

Reason for the proposed changes:

Applicant's invention allows for the chaining of contexts such that the task switcher does not sequentially returns to the last task saved or the order stored but allows requestors to select or dictate which task is restored. Stating the claims in the present form reads on both concepts of 1) ordered restoring and 2) arbitrating restoring.

Therefore, the claims are directed toward the arbitrating restoring which would not be taught by the reference applied or by any other reference previously cited.

Conclusion

7.   Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lewis A. Bullock, Jr. whose telephone number is (571) 272-3759. The examiner can normally be reached on Monday-Friday, 8:30 a.m. - 5:00 p.m..
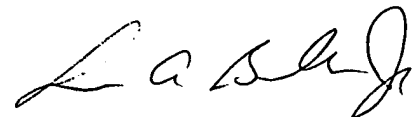
If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Meng An can be reached on (571) 272-3756. The fax phone number for the

organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the

Patent Application Information Retrieval (PAIR) system. Status information for

published applications may be obtained from either Private PAIR or Public PAIR.

Status information for unpublished applications is available through Private PAIR only.

For more information about the PAIR system, see http://pair-direct.uspto.gov. Should

you have questions on access to the Private PAIR system, contact the Electronic

Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a

USPTO Customer Service Representative or access to the automated information

system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

January 5, 2008

LEWIS A. BULLOCK, JR.
PRIMARY EXAMINER